Composing 1120 Return Transmission Files – An Overview

Release No: 1.0 Draft
Date: September 25, 2002

Composing 1120 Return Transmission Files - An Overview

Revision History

Revision Number	Revision Date	Summary of Changes	Changes marked
1.0 Draft	09/25/02	Release 1.0 Draft	

Table of Contents

1. Introduction

This document is intended for the software developers who compose the corporate return tax data in XML format as per the XML Schemas published by the IRS. It documents the structure of the corporate return, the transmission file, and general considerations about using the XML schemas. It provides guidance to the EROs (Electronic Return Originators) on composing the return data (structure), and to the transmitters on composing the transmission file (structure) for submission to the IRS.

This document is in draft stage and will be embellished over time.

The user's cooperation is requested in developing a quality document that provides guidance on the use of XML Schemas to compose the return data and transmission files. If you notice errors (typographical, technical, or usage) or if you have any suggestions and/or comments, please let us know. Please email your comments to 1120@irs.gov

2. Transmission File Structure

The transmission file is a MIME (Multipurpose Internet Mail Extensions) multi-part document that conforms to "SOAP 1.1 with attachments" standard. The first part of the multi-part document is the SOAP envelope and remaining parts are SOAP attachments. MIME boundaries separate the parts in the multi-part document. The SOAP envelope maintains transmission level information, and each SOAP attachment is a corporate return.

The SOAP envelope consists of a SOAP header and a SOAP body. The SOAP header, also referred to as the *transmission header* in the Modernized e-File system, contains information about the transmitter and the transmission. The SOAP body, also referred to as the *transmission manifest*, contains a list of all SOAP attachments in the transmission file.

Each SOAP attachment in the transmission file is MIME multi-part document that contains data for a return. The return consists of one or more MIME parts. The first part is always the return data, and the remaining parts are either returns of subsidiary corporations (in a consolidated return) or non-XML attachments, also referred to as 'binary attachments.

The structure of the transmission file is shown below followed by explanations of the header fields and whether or not they are required. The text to the right side of the brackets explains the purpose of the header fields. Note that the data (structure) for the first return in the transmission file is indented for easy reading and is in blue color.

Note that the values in brackets e.g., "<MIME1120Boundary>" need to be replaced by the application composing the file.

Structure of the transmission file:

MIME-Version: 1.0	required verbatim
Content-Type: Multipart/Related; boundary=<MIME1120Boundary>; type="text/xml"	*Multi-part content header*
Content-Description: This is a sample structure of a transmission file.	Description header
EFile-ContentType: 1120	custom e-File header
--<MIME1120Boundary>	separates parts in trans file
Content-Type: "text/xml"; charset=UTF-8	*body part header*
Content-Transfer-Encoding: 8bit	for the SOAP envelope
Content-Location: <Envelope1120>	
(…SOAP Envelope goes here… contains a SOAP Header and a SOAP body)	
--<MIME1120Boundary>	2nd part, a return, starts here
Content-Type: Multipart/Related; boundary=<Return001PartBoundary>; type="text/xml"	Multi-part content
Content-Location: <01000020020860000001>	header for the
Content-Description: data for the first return (one or more parts) follows.	return
--<Return001PartBoundary>	separates parts in the return
Content-Type: **text/xml**; charset=UTF-8 (parent corp's return)	*body part header*
Content-Transfer-Encoding: 8bit	for the **parent**
Content-Location: <ReturnData001>	corporation's
(Return (XML) data for the parent return goes here)	parent return data

--<Return001PartBoundary> Content-Type: **text/xml**; charset=UTF-8 Content-Transfer-Encoding: 8bit Content-Location: <SubsidiaryReturn001> Content-Description: The subsidiary return is just a part in the multi-part parent return	separates parts in the return body part header for the **subsidiary**
(Return (XML) data for the subsidiary return goes here)	subsidiary return data
(…returns for additional subsidiaries, if any, go here ..)	
--<Return001PartBoundary> Content-Type: **application/pdf** Content-Transfer-Encoding: Binary Content-Location: <BinaryAttachment001> Content-Description: Scanned image of the officer's signature (form 8453)	separates parts in return body part header for **non-xml** data
(Binary attachment file goes here)	PDF file
(… other non-XML (binary) attachments , if any, go here)	
--<Return001PartBoundary>--	end of return001
--<MIME1120Boundary> Content-Type: Multipart/Related; boundary=<Return002PartBoundary>; type="text/xml" Content-Location: <01000020020860000002> Content-Description: This is the second return in the transmission file.	Begin 2nd return Multi-part header for the second return in the transmission file
--<Return002PartBoundary> (…other parts in the return, if any, go here)	all data (parts) second return go here
--<Return002PartBoundary>--	end of 2nd return
--<MIME1120Boundary>	Begin 3rd part (return)
(… other returns go here…)	
--<MIME1120Boundary>--	end of parts in trans file

2.1. Header fields used in the transmission file

The *MIME content headers,* or simply the *content headers* are used to describe the contents of MIME parts. The *multi-part content header* (Content-Type="Multipart/Related") specifies a boundary value that separates MIME parts in a MIME multi-part structure. Other header fields provide additional information about the MIME part. There is one *multi-part content header* for each MIME multi-part structure.

Since each return is a MIME multi-part structure, it contains one *multi-part content header*. The value in the 'boundary' parameter of this content header separates parts (return XML data, and non-XML data) in the return.

Since the transmission file is also a MIME multi-part structure, it contains one *multi-part content header*. The value in the 'boundary' parameter of this content header separates parts- the SOAP envelope and the SOAP attachments in the transmission file.

MIME-Version header field
The MIME-Version header field is **required** at the top level of a message in the transmission file. It is not required for each body part of a multipart document. It must be included with the following verbatim text:

MIME-Version: 1.0

Content-Type header field
The Content-Type header field is **required**. The Content-Type header field describes the nature of the data in the MIME part by giving media type and subtype identifiers.

The value of the Content-Type header field in a multi-part message structure must be set to "Multipart/Related" and a value must be provided for both the 'boundary' parameter and the 'type' parameter. It must appear as follows:

Content-Type: Multipart/Related; boundary=<MIME1120Boundary>; type="text/xml"

The 'boundary' parameter separates parts in the multi-part document- it separates the returns in a transmission file, and separates return data from the non-XML attachments (and subsidiary returns) in a return. The value for the 'boundary' parameter ("MIME1120Boundary" above) is to be created by the application composing the transmission file. The 'type' parameter indicates the content type of the multipart/related object. It must have the value "text/xml" as per SOAP 1.1 specification.

The value of the Content-Type header field is set to either "text/xml" when describing an XML part, or is set to "application/pdf" when describing a non-XML part.

When it is included in the *body part header* of the SOAP envelope (in the transmission file), or the return data (in the return), or the subsidiary return (in the return), its value must be set to "text/xml" as shown below:

Content-Type: "text/xml"; charset=UTF-8

When Content-Type header field is included in the *body part header* of a 'binary attachment', its value must be set to "application/pdf" as shown below:

Content-Type: "application/pdf"

The 'charset' parameter must be set to the value "UTF-8" when the value of the header is "text/xml". This parameter is not required when the value of the header is "application/pdf".

The names of the type, subtype, and parameters are not case sensitive, however, the parameter values are.

Content-Description header field
The Content-Description header field is **optional**. It is not processed by the Modernized e-File system.

Content-Transfer-Encoding header field
The Content-Transfer-Encoding header field is **required** for MIME parts whose Content-Type header has the value "text/xml" or "application/pdf". Its value must be set to "8Bit" for MIME parts

of Content-Type "text/xml", and it must be set to "binary" for MIME parts of Content-Type "application/pdf".

It is not required when the Content-Type header field has any other value e.g., "Multipart/Related". However, If it is included with the MIME part that has the Content-Type set to "Multipart/Related", it is not permitted to have any value other than "7Bit", "8Bit", or "binary".

Content-Location header field
The Content-Location header field is **required**. It specifies an URI that labels the content of the body part in whose heading it is placed. References are made to these labels from the root body part of the multipart/related MIME message.

In the transmission file, there are references to these labeled body parts (returns) from the transmission manifest (the root body part).

In the return, also a multipart/related MIME structure, there are references to the labeled body parts (attachments) from inside the return structure.

The URIs in the Content-Location headers SHOULD be globally unique. The e-File system requires that the URIs within the transmission file be unique i.e., each return have a unique ReturnID (which is its Content-Location), AND, the URIs within the return be unique i.e., each MIME part within a return have a unique value for the Content-Location header. It does not require that Content-Location values inside a return be unique within the transmission file; they just need to be unique within the return.

Efile-ContentType header field
The Efile-ContentType header field is **required** in the transmission file. It specifies the 'type' of returns included in the transmission file. It has been created for use by the Modernized e-File system. It is not a part of the standard MIME Verion 1.0 specification. The transmitter sets the value of this header field. It contains a scaler value. At this time, the only valid value for this header field is "1120" which indicates that the file contains corporate income tax returns (either 1120, or 1120S) as shown below:

EFile-ContentType: 1120 (the transmission file contains corporate returns only)

3. What does the transmitter do when composing the transmission file?

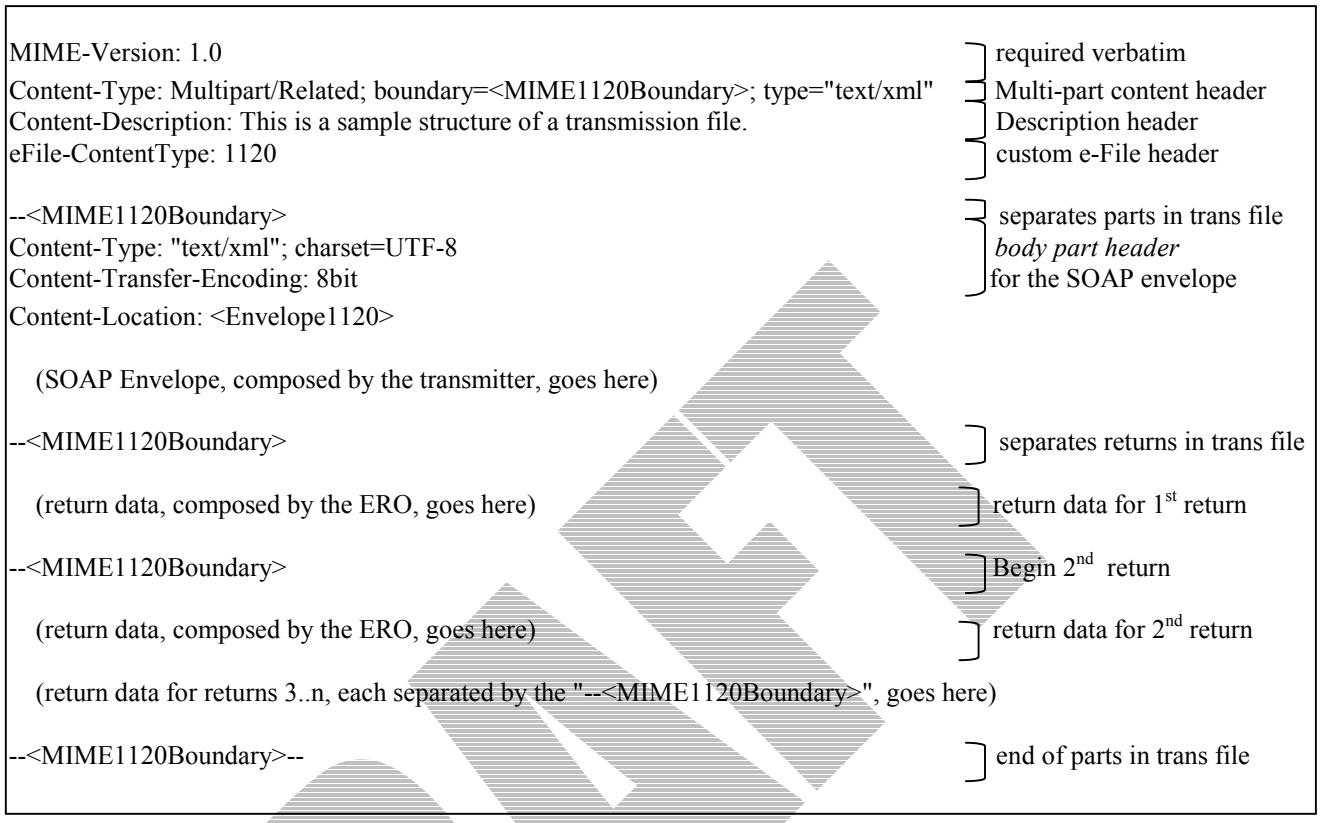
The transmitter is an IRS authorized *e-file* provider that transmits electronic tax returns to the IRS. The transmitters receive the electronic data for the returns from the EROs (Electronic Return Originators). This section describes the structure of the transmission file that the transmitter must follow and general guidance on composing the transmission file.

The transmission file is a MIME multi-part structure that conforms to the "SOAP 1.1 with attachments" standard. The first part of the multi-part document is the SOAP envelope and remaining parts are SOAP attachments. MIME boundaries separate the parts in the multi-part MIME structure. The SOAP envelope maintains transmission level information, and each SOAP attachment is a corporate return.

The SOAP envelope consists of a SOAP header and a SOAP body. The SOAP header, also referred to as the *transmission header* in the Modernized e-File system, contains information about the transmitter and the transmission. The SOAP body, also referred to as the *transmission manifest*, contains a list of all SOAP attachments in the transmission file. The transmitter is responsible for composing the transmission header. The transmitter needs to know the ReturnID of each return that is included in the transmission file. The ReturnID is generated and provided to the transmitter by the ERO.

Each SOAP attachment, a MIME part in the transmission file, is itself a MIME multi-part structure that contains return data for a corporation. The ERO is responsible for composing the return data for the corporation.

The general structure of the transmission file, a MIME multi-part structure, is depicted below. It is followed by explanations where appropriate. Note that the values in angled brackets e.g., "<Return001PartBoundary>" need to be replaced by the application composing the MIME structure for the return.



General structure of the transmission file, a MIME multi-part structure

3.1. MIME content headers and MIME parts in the transmission file

The *MIME content headers,* or simply the *content headers* are used to describe the contents of MIME parts. The *multi-part content header* (Content-Type="Multipart/Related") specifies a boundary value that separates parts in a MIME multi-part structure. Since the transmission file is a MIME multi-part structure, there is one *multi-part content header* at the top of the transmission file. The transmitter creates this *multi-part content header*. The 'boundary' parameter of this header specifies a value that separates the parts in the multi-part transmission file.

The transmission file contains two kinds of parts- the SOAP envelope (also referred to as the *transmission envelope*), and SOAP attachments- each a corporate return. The transmitter composes the SOAP envelope. The ERO's software composes SOAP attachments (the returns) and provides to the transmitter for inclusion in the transmission file.

The transmission envelope contains two components- a transmission header and a transmission manifest. The transmission header contains information about the overall transmission. The transmission manifest is a list of all returns included in the transmission file. The location (the *Content-Location* header field value) of each return is specified in the transmission manifest. This location is the same as the ReturnID of the return, and is provided to the transmitter by the ERO's software.

eFile-ContentType header in the transmission file

The eFile-ContentType is a custom MIME header field that specifies the 'type' of returns included in the transmission file. It has been created for use by the Modernized e-File system. It is not a part of the standard MIME Verion 1.0 specification. The transmitter sets the value of this header field. It contains a scaler value. At this time, the only valid value for this header field is "1120" which indicates that the file contains corporate income tax returns (either "1120" or "1120S").

The first MIME part in the transmission file
The first MIME part in the transmission file is always the SOAP envelope. The SOAP envelope is an XML document that conforms to the SOAP schema described in the soap.xsd file. It is composed by the transmitter.

The MIME parts 2..n in the transmission file
The MIME parts 2..n each represent the return data for a corporation. These (return data MIME parts) are composed by the ERO's software and provided to the transmitter for inclusion in the transmission file. The MIME parts are separated by the boundary value created by the transmitter and specified by the 'boundary' parameter in the Content-Type header field.

4. What does the ERO do when composing the return data?

The ERO (Electronic Return Originator) is the authorized *e-File* provider that originates the electronic submission of an income tax return using approved *e-file* software. The EROs may originate the electronic submission of income tax returns they either prepared or collected from taxpayers. This section describes the structure of an electronic return that the ERO's software must follow when composing the return data.

The electronic return is a MIME multi-part structure. The ERO is required to create the return data per the XML Schema for the return type (1120, or 1120S), and enclose the return data into the MIME structure. The return data can be made up of one or more MIME parts. Each part contains either the return data or a non-XML PDF file. In a consolidated return, the return data for the parent corporation is enclosed in one MIME part, the return data for each of the subsidiary corporations is enclosed in separate MIME parts (one for each subsidiary corporation) and each non-XML data (PDF) file is enclosed into its own MIME part.

The general structure of the return, the MIME multi-part structure, is depicted below followed by explanations where appropriate. Note that the values in angled brackets e.g., "<Return001PartBoundary>" need to be replaced by the application composing the MIME structure for the return.

General structure of a return, a MIME multi-part structure

4.1. MIME headers and MIME parts in the return

The *MIME content headers,* or simply the *content headers* are used to describe the contents of MIME parts. The *multi-part content header* (Content-Type="Multipart/Related") specifies a boundary value that separates MIME parts in a MIME multi-part structure. This section describes the *content headers* and MIME parts that make up the return. For an explanation of the header fields and whether or not they are required, please see the Chapter titled "Transmission File Structure".

The MIME Multi-part content header in the return
Since the return is a MIME multi-part structure, there is one *multi-part content header* (Content-Type="Multipart/Related") at the top of the return. This header is composed by the ERO's software. The 'boundary' parameter of this header specifies a value that separates the parts in the

multi-part return document. Other content headers provide additional information about the MIME part.

The Content-Location header in the return
The Content-Location header field provides a label for the MIME part.

When Content-Location header field is included in the message header of the return i.e., when Content-Type is set to "Multipart/Related', then its value **must** be the same as the ReturnID of the return.

When Content-Location header field is included in the body part header for any part in the return i.e., when the Content-Type is set to either "text/xml" or "application/pdf", it can have any value so long as the value is distinct from all other values used for other Content-Location header fields in the return.

The ERO provides the ReturnID along with the return data (MIME multi-part) structure of each return to the transmitter. The transmitter uses the ReturnID to create the transmission manifest that enumerates (and points to) each return in the transmission file. Please see the Chapter titled "What does the transmitter doe when composing the transmission file" for details on how the transmission file is composed.

The first MIME part in the return
The first MIME part in the return is always the return data (content-type="text/xml") for the corporation's return (1120 or 1120S). The MIME parts for the return data for subsidiary corporations, if any, follow it.

The MIME parts 2..n in the return
If the return contains data for subsidiary corporations and/or contains non-XML data (PDF files), then the return contains more than one MIME part. The return data (MIME parts) for the subsidiary corporations (Content-Type="text/xml") are all placed together following the parent corporation's MIME part. The MIME parts for the non-XML (PDF) data (Content-Type=application/pdf) are all placed together following the last MIME part for the subsidiary return, if any.

5. General philosophy on data elements within the return

In general, most data elements in the schemas for each form, schedule, and supporting document have been declared optional. Most of the required elements are in the return header schema. The return header schema contains identifying information about the corporation filing the return, the officer responsible for the data in the return, the preparer, and the preparing firm. Hence there are very few data elements for which you must provided data values. This philosophy of keeping most data elements optional in the schemas is consistent with the way paper returns are filed i.e., the taxpayer and return preparer have the responsibility to provide information as specified by IRS forms, instructions, and regulations.

Also, the content model of the corporate return (both 1120 and 1120S) declares each component-the form, schedule, or the supporting document optional. The only required components are the return header, and the form 1120 (or form 1120S) in a corporate return. All other forms, schedules, and supporting documents are declared optional in the content model. Hence you must provided data for the required elements in the return header and the form 1120 (or 1120S) when filing a corporate return. Data for all other forms, schedules, and supporting documents is Data for all other forms, schedules, and supporting documents is required based on specific rules that will be provided separately from schemas (e,g,, if Form 1120, Box A1 is checked, Form 851 must be attached).

6. How are the schema files physically organized (i.e., what's in each file)?

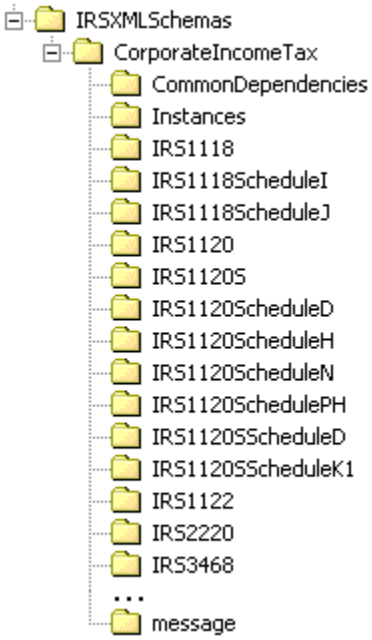


Figure 1 - Abbreviated Screenshot of Schema File Directories

Referring to Figure 1 above, here are the descriptions of the major directories found in the zip file located on the IRS.gov website:

IRSXMLSchemas/ - top level directory of the schemas distribution. It contains **efileTypes.xsd**, which defines the common types used throughout the schemas.

IRSXMLSchemas/CorporateIncomeTax/ - top level directory for corporate income related schemas. It also contains a few schemas defining which forms/schedules are part of an 1120 or 1120S return.

IRSXMLSchemas/CorporateIncomeTax/CommonDependencies/ - contains schemas representing common supporting information attachments referred to by more than one form/schedule including the IRS Corporate Payment.

IRSXMLSchemas/CorporateIncomeTax/Instances/ - contains sample transmission files for returns and IRS responses.

IRSXMLSchemas/CorporateIncomeTax/IRS*/ - contain the schemas specific to the form/schedule, where the name of form is after "IRS" in the name of the directory. For example, schemas for Form 1120 would be found in the directory "IRS1120". Within these "IRS*" directories, the schema for the form/schedule is named identical to the directory name (plus an "xsd" extension), and all other files are the schemas for supporting info documents specific to that form.

IRSXMLSchemas/CorporateIncomeTax/message/ - contains transmission level schema files. These schemas define the transmission header and manifest XML structures as well as base header and acknowledgment structure.

The bulk of the files are the form/schedule schemas along with the common supporting info schemas. These files describe the XML elements and attributes that correspond to the fields and notations on the IRS forms. Generally, the flow of the element definitions follows the flow of the form. Most of the elements are of a type defined in **efileTypes.xsd**. If not, the type is defined within the containing schema. A typical example of a schema element definition is shown below. It defines an element, <CostOfGoodsSold>, which corresponds to an amount field on the form.

```
<!-- Cost of Goods Sold -->
<xsd:element name="CostOfGoodsSold" type="AmountType" minOccurs="0">
      <xsd:annotation>
            <xsd:documentation>
                  <Description>Cost of goods sold</Description>
                  <LineNumber>2</LineNumber>
            </xsd:documentation>
      </xsd:annotation>
</xsd:element>
```

Tables on the forms are defined as repeating groups in the schemas. These repeating groups are actually made up of an element that repeats an unbounded number of times, which wraps around elements defining the column fields. A typical abbreviated example of a repeating group is shown below.

```
<xsd:complexType name="ItemizedOtherAssets">

      <!-- Itemized Other Asset (table) -->
      <xsd:element name="ItemizedOtherAsset" type="OtherAssetsType"
minOccurs="0" maxOccurs="unbounded" />

</xsd:complexType>

<!—Other Assets Info -->
<xsd:complexType name="OtherAssetsType">
      <xsd:sequence>
            <xsd:element name="Description" type="LineExplanationType"
      minOccurs="0" />
            <xsd:element name="Amount" type="AmountType" minOccurs="0"
      />
      </xsd:sequence>
</xsd:complexType>
```

7. How are the supporting materials attached to forms and fields in the return data?

Supporting materials typically are of the following types:

- Forms
- Schedules
- Statements
- Elections
- Attachments
- Payment records
- Notices

Currently, following are the only two permissible formats for these supporting materials in a return.

- XML documents; and
- Non-XML documents, i.e., binary files (only Adobe PDF documents) where permitted.

7.1. Attaching XML documents to forms and fields in a return

Here is a simple example of how an "`Itemized Other Income schedule`" is attached to a field "Other Income" on Form IRS 1120:

```
...
<OtherIncome referenceDocumentId="DEPa1">8880</OtherIncome>
...
```

The Attached document with a `documentId=`**`DEPa1`** (must be unique among other documentIds within that return) should be present in the return. Here is how it should be:

```
<ItemizedOtherIncomeSchedule documentId="DEPa1">
  <ItemizedIncome>
    ...
  </ItemizedIncome>
</ItemizedOtherIncomeSchedule>
```

Every element (a field on a form/schedule or a form/schedule itself) in any return document that may have potential attachments of any return document (form, schedule, or a supporting material) has the following two special attributes.

- `referenceDocumentId` attribute: The attribute would be of type either `IdType` or `IdListType` depending on whether one or more attachment are possible for that element. This attribute field has the same value of the `documentId` (or documentIds) of the document(s) that is (are) being attached.

- `referenceDocumentName` attribute: This attribute contains `documentName(s)` of all the potential types of attachment references that can be made from that element (could be a field on a form or a form/schedule itself) to other forms, schedules and supporting materials.

On a side note, this attribute (where ever it occurs in the return) is used for IRS internal use only and must not be included in the tax return. However, after the return is received IRS XML parsers will add such attributes to the appropriate elements in the return. Here is an example:

In the XML data to be transmitted by the preparer:

```
<OtherIncome referenceDocumentId="RetDoc010">8887
</OtherIncome>
```

After transmission, IRS XML parser will add:

```
<OtherIncome referenceDocumentId="RetDoc010"
referenceDocumentName="ItemizedOtherIncomeSchedule">8887
</OtherIncome>
```

Every return document (form, schedule or supporting material) which is an XML document has the following two special attributes for its root element.

- `documentName` attribute: is an attribute of type `FIXED` (a string literal type) and has a value equals name of the form, schedule or a supporting document it represents.

  On a side note, this attribute (where ever it occurs in the return) is used for IRS internal use only and must not be included in the tax return. However, after the return is received IRS XML parsers will add such attributes to the appropriate elements in the return. Here is an example:

  In the XML data to be transmitted by the preparer:

  ```
  <IRS1120 documentId="RetDoc010">
  ```

  After transmission, IRS XML parser will add:

  ```
  <IRS1120 documentId="RetDoc010" documentName="IRS1120">
  ```

- `documentId` attribute: The purpose of this attribute is to identify it uniquely within the context of the whole return. Tax return preparer's software is responsible for generating a unique id of `idType` defined in `efileTypes.xsd` for each of the return documents. Here are some examples:

  - `<IRS1120 documentId="DOC0001">`
  - `<IRS1120ScheduleD documentId="0020.2225">`
  - `<DualConsolidatedLossesStatement documentId="ABC:002.XY">`

See the sample return shown in the next section.

7.2. Sample return with XML documents as attachments

```
<xml version="1.0" encoding="UTF-8"?>
<CorporateReturn>
<ReturnHeader binaryAttachmentCount="0" version="1.2">
   <ReturnId>01000020020860000001</ReturnId>
   <Timestamp>2002-03-26T13:00:05-05:00</Timestamp>
      ...
      ...
</ReturnHeader>
<ReturnData documentCount="22">

      <!-- Example of (1) element with supporting document "Itemized
Other Income Schedule" and (2) form with three supporting documents attached -
```

```
One  “Stock Ownership Foreign Corp Statement” and two “Dual Consolidated Losses
Statements” -->
   <IRS1120 documentId="DOC0001" referenceDocumentId="DEPc1 DEPd1 DEPd2"
>
      ...
      ...
       <!--  Element with supporting document, “Itemized Other Income Schedule” -->
      <OtherIncome referenceDocumentId="DEPa1">8870</OtherIncome>
      ...
      ...
   </IRS1120>
   ...
   ...
   <!--  Supporting document attached to an element in Form 1120 -->
   <ItemizedOtherIncomeSchedule documentId="DEPa1">
      <ItemizedIncome>
       ...
       ...
      </ItemizedIncome>
   </ItemizedOtherIncomeSchedule>
   ...
   ...
   <!--  Supporting  document attached to Form 1120 -->
   <StockOwnershipForeignCorpStmt documentId="DEPc1">
      ...
      ...
   </StockOwnershipForeignCorpStmt>
   ...
   ...
   <!--  Supporting  documents attached to Form 1120 -->
   <DualConsolidatedLossesStatement documentId="DEPd1">
      <Statement>Statement One</Statement>
   </DualConsolidatedLossesStatement>

   <DualConsolidatedLossesStatement documentId="DEPd2">
      <Statement>Second Statement</Statement>
   </DualConsolidatedLossesStatement>
   ...
   ...
</ReturnData>
</CorporateReturn>
```

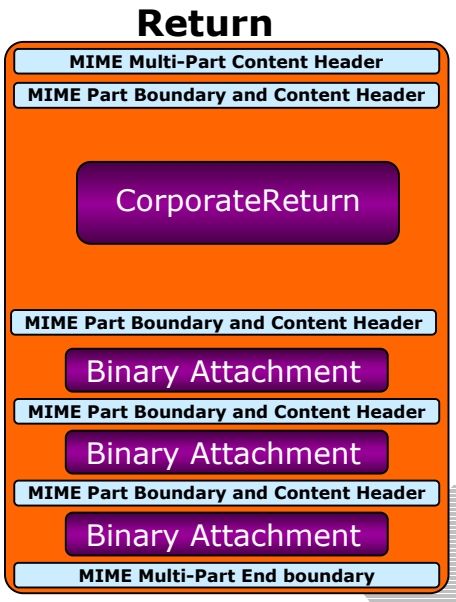
7.3. Attaching non-XML documents, i.e., binary (PDF) files to the return

Binary files are attached differently than supporting XML documents. The actual attachments are included in the return (not with in the `CorportateReturn` element) in the form of a MIME parts as opposed to elements inside `ReturnData` element.

NOTE: Currently, the attachment related attributes (`referenceAttachmentId` and `referenceAttachmentName`) are not used in any of the schemas since binary attachments to fields/forms have not been identified to date. Therefore, here is a **hypothetical** example of how an “`Itemized Other Income PDF File`” is attached to a field, “`Other Income`” **:**

```
…
<OtherIncome referenceAttachmentId="PDF0001">9970</OtherIncome>
…
```

The attached binary PDF document must be included as a MIME part in the Return. The content location for the MIME part must equal "**PDF0001**". See figure below and sample in the next section.



In addition, the attribute `binaryAttachmentCount` of element `ReturnHeader` in `ReturnHeader1120x.xsd` must have the correct count of all the binary attachments included in the return. See the sample return in the next section.

7.4. Segment of a transmission file sample showing an individual return with its binary attachment

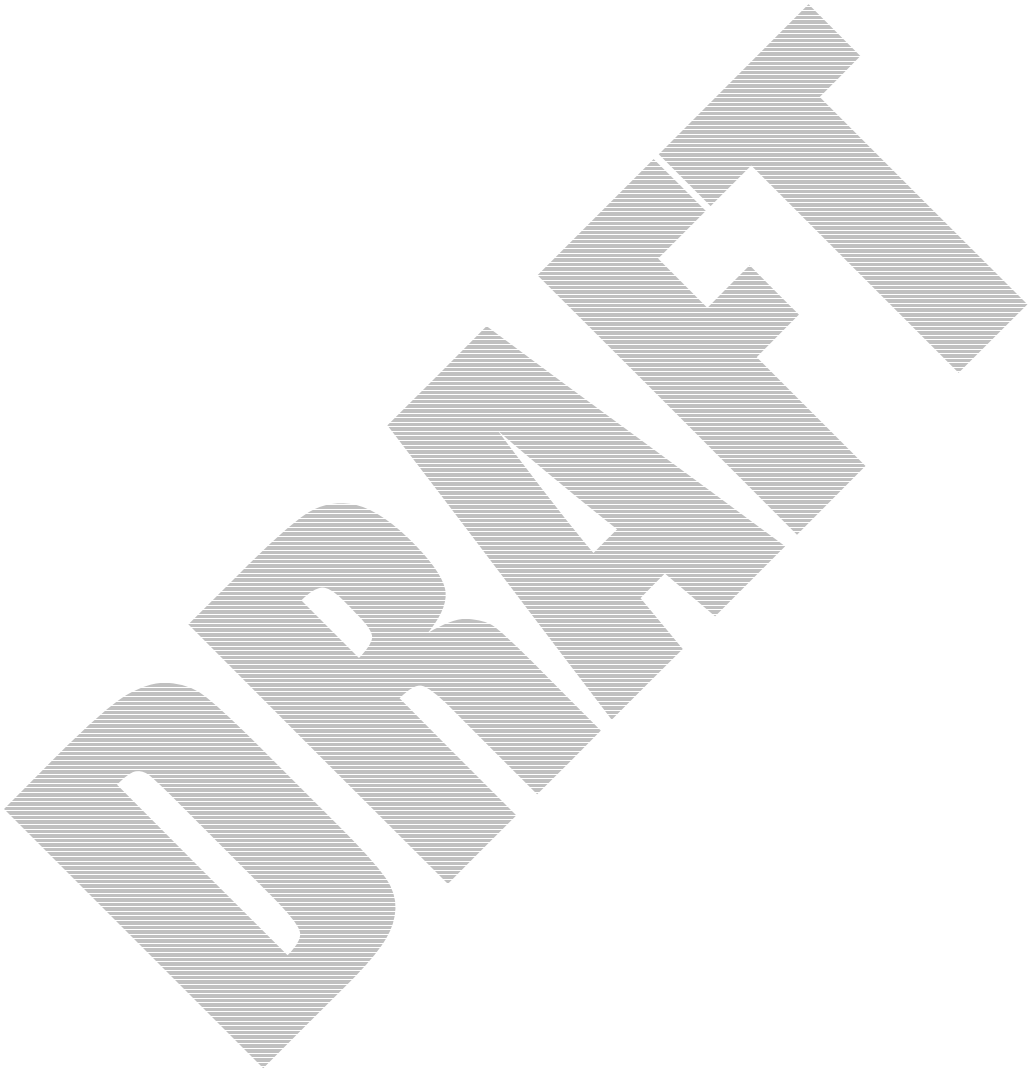
```
--MIME1120Boundary
Content-Type: Multipart/Related; boundary=MIME1120Return0001Boundary;
type=text/xml
Content-Location: Return0001

--MIME1120Return0001Boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-Location: 01000020020860000001

<CorporateReturn>
<ReturnHeader binaryAttachmentCount="1" version="1.2">
     ...
</ReturnHeader>
<ReturnData documentCount="11">
   <IRS1120 documentId="DOC0001">
      ...
      <!--  Element with a PDF document as an attachment -->
      <OtherIncome referenceAttachmentId="PDF0001">9970</OtherIncome>
      ...
   </IRS1120>
   ...
</ReturnData>
```

```
</CorporateReturn>

--MIME1120Return0001Boundary
Content-Type: application/pdf
Content-Transfer-Encoding: Binary
Content-Location: PDF0001
...
...
--MIME1120Boundary
```



8. How do I validate my return against the XML schemas?

Here is a high-level content model of `CorporateReturn` XML document:

```
<xml version="1.0" encoding="UTF-8"?>

<!-- Either Return1120.xsd or Return1120S.xsd for Corporate Return -->
<CorporateReturn>

   <!-- ReturnHeader.xsd for Return Header -->
   <ReturnHeader binaryAttachmentCount="0" version="1.2">
   </ReturnHeader>

   <!-- ReturnData1120.xsd or ReturnData1120S.xsd for All Forms/Fields,
schedules and XML Attachments -->
   <ReturnData documentCount="22">
      <!-- IRS1120.xsd, XML Schema for Form IRS 1120 -->
      <IRS1120 documentId="DOC0001">
         ...
         <!-- Form Fields -->
         <OtherIncome referenceDocumentId="DEPa1">8770</OtherIncome>
         ...
      </IRS1120>
      ...
      <!-- XML Schema for a Dependency, Itemized Other Income Schedule-->
      <ItemizedOtherIncomeSchedule documentId="DEPa1">
         ...
      </ItemizedOtherIncomeSchedule>
      ...
   </ReturnData>

</CorporateReturn>
```

Here is a brief description of above content model:

- A `CorporateReturn` (a corporate tax return) is the root element in XML schema `Return1120.xsd` or `Return1120S.xsd` whichever is appropriate.
- A `CorporateReturn` comprises of two children elements. A `ReturnHeader` based on XML schema `ReturnHeader1120x.xsd`. A `ReturnData` element based on either XML schema `ReturnData1120.xsd` or `ReturnData1120S.xsd` whichever is appropriate.
- A `ReturnHeader` contains elements common across all the individual return documents, e.g., taxpayer name, EIN, address, etc.
- Each individual return document is either a form or schedule or supporting material and has a separate XML schema defined for it.
- All return documents (forms, schedules and supporting materials) are contained within a `ReturnData` XML structure based on XML schema `ReturnData1120.xsd` or `ReturnData1120S.xsd` whichever is appropriate.
- A return may also include binary attachments, see Chapter titled “How are the supporting materials attached to forms and fields in the return data?” for details.

Below are some XML resources regarding XML schemas and software tools and parsers (these resources are provided for information only—the IRS is not endorsing any product)

- W3C XML Home Page: `http://www.w3.org/XML/`

- W3C XML Schema Home Page: `http://www.w3.org/XML/Schema`
- XML Spy: `http://www.xmlspy.com/`
- Apache Xerces parser toolkit: http://xml.apache.org/
- Microsoft XML Core Services toolkit: http://msdn.microsoft.com/downloads/sample.asp?url=/MSDN-FILES/027/001/766/msdncompositedoc.xml
- You may chose any third party parser toolkit or use your own.

8.1. Validating a single return document

As stated earlier, each individual return document (form, schedule, supporting material etc) has its own XML schema defined. Therefore, one does not need to compose the whole return XML document (`Return1120.xsd` or `Return1120S.xsd`) in order to validate that singe return document (a form, schedule, a supporting material).

In other words, one may simply assemble data pertaining to that individual return document (a form, schedule, supporting material) and immediately validate it using its own XML schema.

Here is an example of how to validate a single Return Document, "Itemized Other Income Schedule" XML document.

- Assemble all the elements needed for the `ItemizedOtherIncomeSchedule.xsd` document as follows:

```
<ItemizedOtherIncomeSchedule documentId="ABC00010" >
  <ItemizedIncome>
     <IncomeType>Trade</IncomeType>
     <Amount>22330</Amount>
     ...
     <PartnershipAmount>11120</PartnershipAmount>
  </ItemizedIncome>
</ItemizedOtherIncomeSchedule>
```

- Add the following to the above (text and other required attributes to the root which are shown in **bold** in the example below) to make a stand-alone XML document pointing to appropriate XML schema file. Here is a sample:

```
<?xml version="1.0" encoding="UTF-8"?>
<ItemizedOtherIncomeSchedule xmlns="http://www.irs.gov/efile"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.irs.gov/efile
C:\DevViews\vtadakam_Schemas_R1\SchemasAndStylesheets\IRSXMLSchema
s\CorporateIncomeTax\IRS1120\ItemizedOtherIncomeSchedule.xsd"
documentId="ABC00010">
  <ItemizedIncome>
     <IncomeType>Trade</IncomeType>
     <Amount>22330</Amount>
     ...
     <PartnershipAmount>11120</PartnershipAmount>
  </ItemizedIncome>
</ItemizedOtherIncomeSchedule>
```

Now, validate the above XML document using your own XML parsers/validators.

8.2. Validating the whole return

Following are the two ways one could validate a return against the xml schemas.

(1) One-step approach

Prepare all individual XML documents for the return (e.g., ReturnHeader, forms, schedules, and supporting materials, etc.), and then assemble and validate against the appropriate schema, `Return1120.xsd` or `Return1120S.xsd`.

(2) Multi-step approach
There are several different ways one could do multi-step validation of the return. Here is one way:

- Prepare a `ReturnHeader` document and validate against its schema, `ReturnHeader1120x.xsd`.
- Prepare each individual return document (forms, schedules, and supporting materials) and validate against its schema.
- Assemble all the return documents into `ReturnData` and validate against the respective schemas, `ReturnData1120.xsd` or `ReturnData1120S.xsd`.
- Assemble `ReturnHeader` and `ReturnData` into Corporate`Return` and validate against the appropriate schema, `Return1120.xsd` or `Return1120S`.
- This completes the whole return process validation.

Here is another way which is a variation from the procedure shown above:

- Prepare a `ReturnHeader` document. One may or may not validate against its schema, `ReturnHeader1120x.xsd` before proceeding.
- Prepare each individual return document (Forms, schedules, and supporting materials). One may or may not validate against its schema before proceeding.
- Assemble all the return documents into `ReturnData`. One may or may not validate against the respective schemas, `ReturnData1120.xsd` or `ReturnData1120S.xsd` before proceeding.
- Assemble `ReturnHeader` and `ReturnData` into `CorporateReturn` and validate against the appropriate schema, `Return1120.xsd` or `Return1120S.xsd`.
- This completes the whole return validation process.

Choose the procedure that is convenient and best fits your needs.

8.3. Validating the transmission envelope including its contents

The transmission file is a MIME multi-part document that conforms to "SOAP 1.1 with attachments" standard. The first part of the multi-part document is the SOAP envelope and remaining parts are SOAP attachments. MIME boundaries separate the parts in the multi-part document. The SOAP envelope maintains transmission level information, and SOAP attachments are a corporate returns included in the transmission file.

The SOAP envelope consists of a SOAP header and a SOAP body. The SOAP header, also referred to as the *transmission header* in the Modernized e-File system, contains information about the transmitter and the transmission. The SOAP body, also referred to as the *transmission manifest*, contains a list of all SOAP attachments in the transmission file.

Validation of SOAP envelope or transmission envelope including its contents consists of the following steps.

- Validate the SOAP envelope or transmission envelope XML instance against the SOAP schema, SOAP.xsd. The standard SOAP schema has been used without modification, http://schemas.xmlsoap.org/soap/envelope/.

- Transmission header SOAP structure must consist of a single element, `TransmissionHeader`. Validate `TransmissionHeader` element against `efileMessage.xsd`.

  See section "Validating a single return document" for details on how to validate an individual XML document (in this case, transmission Header XML document).

- Transmission body SOAP structure must consist of a single element, `TransmissionManifest`. Validate `TransmissionManifest` element against `efileMessage.xsd`.

  See section "Validating a single return document" for details on how to validate an individual XML document (in this case, transmission manifest XML document).

8.3.1. Sample SOAP envelope with TransmissionHeader and TransmissionManifest

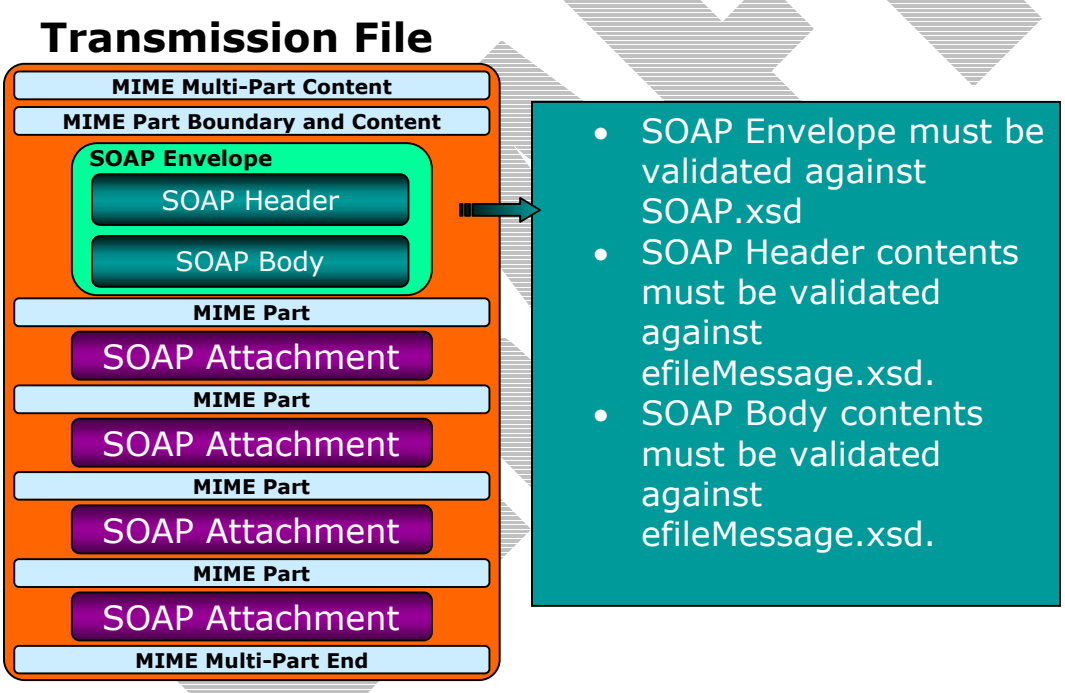
Below is a sample SOAP Envelope based on `SOAP.xsd`. Note that the contents for the `SOAP:Header` and `SOAP:Body` are based on the `TransmissionHeaderType` and `TransmissionManisfestType` defined in `efileMessage.xsd`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!– SOAP:Envelope must be validated against SOAP.xsd -->
<SOAP:Envelope xmlns="http://www.irs.gov/efile"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:efile="http://www.irs.gov/efile"
xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
../message/SOAP.xsd http://www.irs.gov/efile
../message/efileMessage.xsd">
   <SOAP:Header>
      <!– efile:TransmissionHeader must be validated against
efileMessage.xsd -->
     <efile:TransmissionHeader version="1.2">
       <TransmissionId>TID01</TransmissionId>
       <Timestamp>2002-03-27T14:30:00-05:00</Timestamp>
       <Transmitter>
          <ETIN>24317</ETIN>
       </Transmitter>
       <ProcessType>P</ProcessType>
     </efile:TransmissionHeader>
   </SOAP:Header>
   <SOAP:Body>
      <!– efile:TransmissionManifest must be validated against
efileMessage.xsd -->
     <efile:TransmissionManifest count="4">
       <Reference contentLocation="01000020020860000001"
electronicPostmark="2002-03-27T08:20:00-05:00" returnType="1120" />
```

```
        <Reference contentLocation="01000020020860000002"
electronicPostmark="2002-03-27T08:20:10-05:00" returnType="1120" />
     <Reference contentLocation="01000020020860000003"
electronicPostmark="2002-03-27T08:20:20-05:00" returnType="1120" />
     <Reference contentLocation="01000020020860000004"
electronicPostmark="2002-03-27T08:20:30-05:00" returnType="1120" />
     </efile:TransmissionManifest>
   </SOAP:Body>
</SOAP:Envelope>
```

8.3.2. SOAP envelope and transmission file

The figure below shows how the SOAP envelope relates to the transmission file.



9. What happens when the return data does not conform to the schemas?

The return data in the transmission file is validated to make sure it is free of errors before it is accepted. Examples of such errors include a missing EIN in the Return Data, or the EFIN in the Return Header is not present in the IRS Database, or an attachment in the Return is missing. Return Data Validation consists of

- Validating the Return Data against the Schema published by the IRS (Schema Validation)
- Validating the Return Data against IRS Databases (Database Validation)
- Validating the Return Data to make sure they conform to the relevant business rules (Business Rule Validation)

When errors are found in the return, they are reported back to the sender in the acknowledgement file. An example of a sample acknowledgement file is provided in Chapter titled "How are data validation errors reported back to the sender?"

Each error generated contains the following information:

- **Path** - (Xpath) to the data element causing the violation
- **Error Category** - Errors are grouped into a small number of categories
- **Error Message** - Rule text or XML validator message
- **Rule Number** - Each rule is identified by a unique rule number.
- **Severity** - 'Reject' or 'Alert'
- **Data value** - Data value causing the violation - when appropriate

Structural errors are generated when the Return Data does not conform to the structure specified in the XML Schema or when the data type, data length or the range of data values is in error. The types of errors that are generated when the Return Data does not conform to the XML Schema fall into one of the following four categories:

- **Invalid Data Type**
  When the data value does not conform to its type definition (e.g., text in a date field).

- **Invalid Data Length**
  When the length of data does not conform to its specification (e.g., only 4 digits provided for SSN).

- **Invalid Data Format**
  When the structure of data does not conform to its specification (e.g., EFIN does not start with the two digit IRS assigned District Office Code).

- **XML Error**
  When the data violates the Schema (Content Model) specification (e.g., missing a required element, or multiple occurrences of an element e.g., Form 1120).

Each business rule is assigned a severity by the IRS. Severity of the error message determines if a return is accepted or rejected. If an error message with severity of 'Reject' is generated, it causes the Return to be rejected. If an error message has a severity of 'Alert', an 'Alert' is sent back to the sender in the acknowledgement file. If there are one or more 'Alerts' but no 'Rejects' in the acknowledgment file, the return is accepted. The system continues to process the return until it is complete or a maximum number of error conditions (determined by the IRS) are reached.

The acknowledgement file generated after the return is processed is parsed by the sender to inform the tax preparer if there are any Rejects or Alerts in their return.

Example below shows the structure of the error message when the return data does not conform to the schema. The Error Message below is the rule text along with the message given by the XML vValidator.

Here an 'XML Error' is generated, because "City" was not provided as part of the preparer firm's address.

Business Rule

> The return (XML documents) must conform to the version of the XML Schema indicated by the 'version' element in the return header.

This business rule covers all conditions related to Schema Validation. When violated, the following data is reported in the Acknowledgement file:

Xpath: /ReturnHeader/PreparerFirm/PreparerFirmAddress/State

Error Category: XML Error

Error Message: The return (XML documents) must conform to the version of the XML Schema indicated by the 'version' element in the return header. *//COMPLEX_E_UNEXPECTED_CONTENT// Unexpected Content "State"; expected "City".*

Rule Number: 000000001
(NOTE: Final rule number format has not yet been determined.)

Severity: Reject

Data Value:

Text in *italics* is provided by the XML Validator (presently XML Validator by TIBCO is used)

10. What happens when the return data (conforms to the schemas but) does not validate against the IRS databases?

When the Return Data conforms to the schema but does not validate against IRS databases or violates one or more business rules, errors are generated. Examples of errors of this kind include supporting information in the Return is missing, a relationship between data elements in the Return is violated, or data in the Return does not validate against IRS databases. Errors of this nature are classified into eight categories given below:

- **Database Validation Error**
  When some data provided in the return must be present in the IRS database, but is not (e.g., SoftwareID in the Return Header must have passed testing for the return type and tax year).
- **Missing Document**
  When a return document is required per some business rule and is not included in the return (e.g., If Form 8586 Line 4 has a non-zero value then one or more Form 8609 must be attached).
- **Multiple Documents**
  When more than the allowable number of documents (per some business rule) are included in the return.
- **Missing Data**
  When data that should have been provided per some business rule is not provided (e.g., If Form 8865 is attached, then Schedule N (Form 1120) Line 2 must have a non-zero value).
- **Incorrect Data**
  When data is syntactically correct, but violates some business rule (e.g., Field nn cannot contain a value more than 15 million).
- **Data Mismatch**
  When the value in two fields should be the same (may be the source is different), but is not (e.g., Form 1120, Line 2 must equal Schedule A (Form 1120) Line 8).
- **Duplicate Condition**
  When a return or transmission is a duplicate of a previously accepted return or transmission.
- **Math Error**
  When the result of some computation is incorrect (e.g., When Form 1120, Line 31 plus (+) Line 33 is less than Line 32h, then Line 31 plus (+) Line 33 minus (-) Line 32h must equal Line 35).

The Error structure remains the same as described for schema validation. The 'Error Message' in this case is the text of the Business rule as shown by the example below:

Example below illustrates the error structure when a 'Data Mismatch' error is encountered:

Business rule Form 8827, Line 8 must equal Form 1120, Schedule J, Line 6e

When violated, the following data is reported in the Acknowledgement file:

Xpath: /ReturnData/IRS8827/MinimumTaxCredit

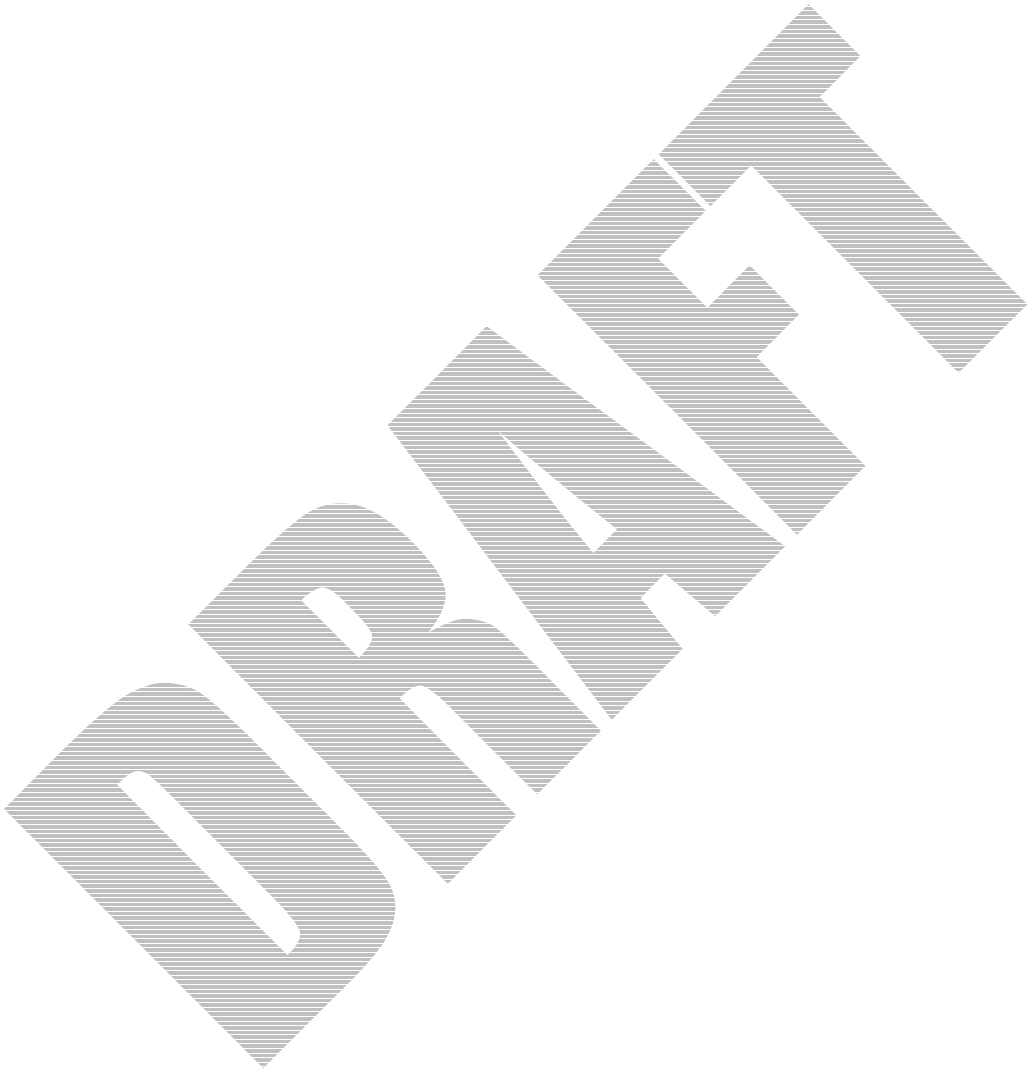
Error Category: Data Mismatch

Error Message: Form 8827, Line 8 must equal Form 1120, Schedule J, Line 6e

Rule Number: 882700001
(NOTE: Final rule number format has not yet been determined.)

Severity: Reject

Data Value: 12345



11. How are the data validation errors reported back to the sender?

The IRS e-File System processes the return completely or until a maximum number of error conditions are reached. If the system finds one or more errors (either rejects or alerts) it reports them back to the sender in an acknowledgement file. The structure of the error element is as given below:

```
<Error>
        <Xpath></Xpath>
        <ErrorCategory></ErrorCategory>
        <ErrorMessage></ErrorMessage>
        <RuleNumber></RuleNumber>
        <Severity></Severity>
        <DataValue></DataValue>
</Error>
```

Example below shows a section of the acknowledgement file when the following business rule is violated:

Rule Number: 112000001 (NOTE: Final rule number format has not yet been determined.)
Rule Text: If Form 1120, Line 26 has a non-zero value, then one "Itemized Other Deductions Schedule" [ItemizedOtherDeductionsSchedule] must be attached.
Severity: Reject

Acknowledgement File

```
<ReturnAcknowledgement xmlns="http://www.irs.gov/efile"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.irs.gov/efile ../message/efileMessage.xsd">
 <ReturnId>01000020020860000017</ReturnId>
 <Errors>
  <Error errorId="1">
    <Xpath>/CorporateReturn/ReturnData/IRS1120/OtherDeductions</Xpath>
    <ErrorCategory>Missing Document</ErrorCategory>
    <ErrorMessage> If Form 1120, Line 26 has a non-zero value, then one "Itemized
                   Other Deductions Schedule" [ItemizedOtherDeductionsSchedule] must
                    be attached
    </ErrorMessage>
    <RuleNumber>112000001</RuleNumber>
    <Severity> Reject </Severity>
    <DataValue></DataValue>
  </Error>
  <Error errorId="2">
        .
  </Error>
</Errors>
 .
 </ReturnAcknowledgement>
```

12. Identifying numbers and their scope

Within the schemas, identifying numbers are defined at the transmission, return, and form levels. These identifying numbers are considered to be key elements/attributes in their containing structures. Their purpose is to identify unique entities, such as a document, organization, person, etc.

Transmission Level Identifying Numbers:

Transmission ID – element <TransmissionId> in the <TransmissionHeader> element definition, this number identifies a unique transmission for the tax year. It is created by the transmitter. It can be up to 30 digits in length, is alphanumeric, and can contain characters ":", ".", and "-". This pattern should allow for a timestamp to be used within the field. This identifying number is also found in the <TransmissionAcknowledgement> element definition.

Transmitter's ETIN – found within the <TransmissionHeader> element definition, this number identifies the unique electronic transmitter. It's a unique 5 digit identification number assigned by the IRS.

Return Level Identifying Numbers:

Return ID – element <ReturnId> in the <ReturnHeader> element definition, this number identifies a unique return within the transmission. It is assigned by the originator. It is 20 digits in length, and allows for ASCII characters. The pattern is: EFIN (6 digits) -> Year (4 digits) -> Julian Day (3 digits) -> Sequence Number (7 alphanumeric). This identifying number is also found in the <ReturnAcknowledgement> element definition.

Software ID – element <SoftwareId> in the <ReturnHeader> element definition, this number identifies the software used to build the return. It's an 8 digit ASCII character field assigned by the IRS.

Originator's EFIN – found within the <ReturnHeader> element definition, this is the originator's Electronic Filing Identification Number. It's a 6 digit numeric field, where the first 2 digits represent a pre-defined IRS district office code. This identifier is assigned by the IRS.

Business's EIN – found within the <ReturnHeader> element definition, this is the Employer Identification Number of the business for which the return is being filed. It's a 9 digit numeric field, where the first 2 digits represent a pre-defined IRS district office code. This identifier is assigned by the IRS.

Preparer's SSN or PTIN – found within the <ReturnHeader> element definition, this is a choice between a person's Social Security Number or Preparer Personal Identification Number. SSN is a 9 digit numeric field, and PTIN is 9 digits, beginning with the letter 'P' followed by 8 numeric digits.

Preparer Firm's EIN – found within the <ReturnHeader> element definition, this is the Employer Identification Number of the firm which prepared the return (if applicable). It is a 9 digit numeric field, where the first 2 digits represent a pre-defined IRS district office code.

Form Level Identifying Numbers:

Software ID – attribute `softwareId` in every top level element in the form/schedule schemas, this number identifies the software used to build the single form/schedule XML instance. It is an 8 digit ASCII character field. If more than one software package is used to compose the return, identify the software ID used to create each form within the return.

Document ID – attribute `documentId` in every top level element in the form/schedule schemas, this number uniquely identifies a single form/schedule XML instance within the return. It can be up to 30 digits in length, and is alphanumeric, plus can contain characters ":", ".", and "-". This pattern should allow for a timestamp to be used within the field. This identifier is assigned by the ERO's software.

Reference Document ID – attribute `referenceDocumentId` found in element definitions where attachments to supporting info documents are made, this number refers to a unique form/schedule XML instance (identified by its `documentId` attribute) within the return. Thus, this attribute's structure is identical to the structure of 'the `documentId` attribute.

Reference Attachment ID – attribute `referenceAttachmentId` found in element definitions where attachments to binary documents are made, this number refers to a unique binary attachment (identified by a `Content-Location` header value) within the MIME multi-parts of a return. This attribute's structure is identical to the structure of the `documentId` attribute.

Reference Attachment Name – attribute `referenceAttachmentName` found in element definitions where attachments to binary documents are made, this string can describe the type of binary attachment (referred to by a `referenceAttachmentId`).

NOTE: Currently, the attachment related attributes (`referenceAttachmentId` and `referenceAttachmentName`) are not used in any of schemas since binary attachments to fields/forms have not been identified to date

13. Consolidated Return Structure

A consolidated return consists of the parent corporation's return and one or more subsidiary corporation's returns. The consolidated return is a MIME multi-part structure (document) that can have multiple parts of type 'text/xml', each containing data for a corporation (parent or the subsidiary). The first MIME part of 'text/xml' type always contains data for the parent corporation and each of the other parts, if any, contain data for a subsidiary corporation. When no data for any subsidiary corporation is provided, only one MIME part of 'text/xml' is found in the return. The non-XML (PDF) attachments, if any, belong to the parent corporation.

The overall structure of the consolidated return is depicted below. The text after the bracket explains parts of structure.

Content-Type: Multipart/Related; boundary=<Return001PartBoundary>; type="text/xml"] Multi-part content
Content-Location: 0100020020860000001 header for the
Content-Description: This is a consolidated return. return

--<Return001PartBoundary>] MIME Part boundary
Content-Type: **text/xml**; charset=UTF-8 (parent corp's return) body part header
Content-Transfer-Encoding: 8bit for the **parent**
Content-Location: <ReturnData001> corporation's return

(Return (XML) data for the parent return goes here)] parent return data

--<Return001PartBoundary>] MIME Part boundary
Content-Type: **text/xml**; charset=UTF-8 (a subsidiary return)
Content-Transfer-Encoding: 8bit body part header
Content-Location: <SubsidiaryReturn001> for the **subsidiary**
Content-Description: The subsidiary return is just a part in the multi-part parent return return

(Return (XML) data for the subsidiary return goes here)] subsidiary return data

(…return data for additional subsidiaries, separated by the MIME part boundary, goes here ..)

--<Return001PartBoundary>] MIME part boundary
Content-Type: **application/pdf** (a PDF attachment)
Content-Transfer-Encoding: Binary body part header
Content-Location: <BinaryAttachment001> for the binary (**non-xml**)
Content-Description: Scanned image of the officer's signature (form 8453) data

(Binary attachment file goes here)] PDF file

(… other non-XML attachments, if any, go here)

--<Return001PartBoundary>--] end of return